REMARKS

Claims 1, 4-8, 11-15, 18-19, 22-28, 30-32, 34-36 and 38-39 are pending in the present application. Applicants respectfully request reconsideration of the application in view of the remarks made herein.

I. Rejections Under 35 U.S.C. § 103

Claims 1, 4-8, 11-15, 18-19, 22-28, 30-32, 34-36 and 38-39 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over *Cavanaugh* (US 5,809,507) in view of *Bannon* (US 5,297,279). The Examiner essentially stated that the combined teachings of *Cavanaugh* and *Bannon* teach or suggest all of the limitations of Claims 1, 4-8, 11-15, 18-19, 22-28, 30-32, 34-36 and 38-39.

Claims 1, 15, 26 and 38 are the independent claims.

Claims 1 and 38 claim, *inter alia*, "generating automatically, by the processor, a database table within the persistent storage structure to store the object instance data." Claim 15 claims, *inter alia*, "generating automatically a database table within the persistent storage structure to store the object instance data." Claim 26 claims, *inter alia*, "automatically generating a database table within the persistent storage structure for storing object instance data."

Cavanaugh teaches marshaling a persistent object attribute value into a marshal buffer, updating the persistent object attribute value while in the marshal buffer, unmarshaling the updated persistent object attribute value from the marshal buffer, and writing the updated persistent object attribute value to a data store (see Abstract). *Cavanaugh* does not teach or

suggest "generating automatically, by the processor, a database table within the persistent storage structure to store the object instance data" (emphasis added) as claimed in Claims 1 and 38 and essentially as claimed in Claims 15 and 26. Consider that the system of *Cavanaugh* includes three groups of classes: a first group defining persistent objects and methods for implementing persistence (224); a second group defining a persistence data storage mechanism (225); and a third group defining a data store (226) into which persistence is written and retrieved (see col. 9, lines 45-50 and FIG. 2). Thus, in *Cavanaugh*, persistent data is stored in the third group of classes, the data store (226) (see also col. 10, lines 34-35). As shown in FIG. 2, the data store (226) does not comprise a database table for storing persistent object instance data. Indeed, the only database tables taught by *Cavanaugh* are m-tables. These m-tables are not located in the data store (226). Further, these m-tables are used to store code which may be used for marshaling data and unmarshaling results (see col. 11, lines 10-17); the m-tables are not used to store object instance data. By way of comparison, Claims 1, 15, 26 and 38 claim a database table located within a persistent storage structure for storing object instance data. Thus, *Cavanaugh* does not teach or suggest all of the limitations of Claims 1, 15, 26 and 38.

Bannon teaches storing persistent objects in a persistent object server (POS server) (54) located within an object-oriented database (OODB) (18) and controlling the POS server (54) with relational database software (RDBMS) (20) located outside of the OODB (see FIG. 2; col. 9, lines 28-37; col. 10, lines 11-15). *Bannon* does not teach or suggest "generating automatically, by the processor, a database table within the persistent storage structure to store the object instance data" (emphasis added) as claimed in Claims 1 and 38 and essentially as claimed in Claims 15 and 26. Indeed, the Examiner does not rely on *Bannon* as teaching this limitation. Thus, *Bannon* fails to cure the deficiencies of *Cavanaugh*.

The combination of *Cavanaugh* and *Bannon* teaches a marshaling process utilizing m-tables to store code which may be used for marshaling data and unmarshaling results and storing persistent objects in a persistent object server (POS server) located within an object-oriented database (OODB) and controlling the POS server with relational database software (RDBMS) located outside of the OODB. The combination does not teach or suggest "generating automatically, by the processor, a database table within the persistent storage structure to store the object instance data" as claimed in Claims 1 and 38 and essentially as claimed in Claims 15 and 26. Accordingly, the combination does not teach or suggest all of the limitations of Claims 1, 15, 26 and 38.

Referring to Claim 26; Claim 26 claims, *inter alia*, "wherein the persistent storage structure is automatically configured to comprise: a persistence module and a storage mapping module for automatically generating a database table within the persistent storage structure for storing object instance data" (emphasis added). In the rejection of Claim 26, the Examiner states that Claim 26 is a system claim corresponding to the method of Claim 1 and is thus rejected for the same reasons as set forth in the rejection of Claim 1 (see page 7 of the Final Office Action). Applicants respectfully submit that implementing a persistence module and a storage mapping module to automatically generate a database table within the persistent storage structure for storing object instance data, essentially as claimed in Claim 26, is not considered in the rejection and is not taught or suggested by either *Cavanaugh* or *Bannon*. Indeed, as discussed above, neither reference discloses generating a database table within a persistent storage structure for storing object instance data, let alone utilizing a persistence module and a storage mapping

module for generating the database table. Thus, the combination of *Cavanaugh* and *Bannon* fails to teach or suggest all of the limitations of Claim 26.

Claim 1 further claims, *inter alia*, "generating automatically, by the processor, an interface within the persistent storage structure, wherein the interface comprises access object classes that are generated automatically by the processor to enable management of object instance data in the persistent storage structure." Claim 15 further claims, *inter alia*, "generating automatically an interface within the persistent storage structure, wherein the interface comprises access object classes that are generated automatically to enable management of object instance data in the persistent storage structure." Claim 26 further claims, *inter alia*, "wherein the persistent storage structure is automatically configured to comprise: an access interface comprising access object classes that are generated automatically to enable management of the object instance data in the persistent storage structure." Claim 38 further claims, *inter alia*, "generating automatically, by the processor, an interface within a persistent storage structure, wherein the interface comprises access object classes that are generated automatically to enable management of object instance data in the persistent storage structure."

Cavanaugh teaches marshaling a persistent object attribute value into a marshal buffer, updating the persistent object attribute value while in the marshal buffer, unmarshaling the updated persistent object attribute value from the marshal buffer, and writing the updated persistent object attribute value to a data store (see Abstract). *Cavanaugh* does not teach or suggest "generating automatically, by the processor, an interface within the persistent storage structure, wherein the interface comprises access object classes that are generated automatically

by the processor to enable management of object instance data in the persistent storage structure" (emphasis added) as claimed in Claim 1 and essentially as claimed in Claims 15, 26 and 38. As stated above, in *Cavanaugh*, persistent data is stored in a data store (226) (see col. 10, lines 34-35 and FIG. 2). *Cavanaugh* uses a marshaling process to manage this stored data. The marshaling process is implemented with m-tables that are used for marshaling data and unmarshaling results. Stated another way, the m-tables in *Cavanaugh* are used to manage persistent data in the data store (226) (e.g., writing data for persistent objects to the data store or retrieving data for persistent objects from the data store) (see col. 11, lines 10-17). As shown in FIG. 2, these m-tables are not located within the persistent data store (226). Thus, by way of comparison, while an interface within the persistent storage structure is utilized to enable management of object instance data in the persistent storage structure in Claims 1, 15, 26 and 38, m-tables located separate and apart from the data store are used to extract data from the data store, manage the data, and return the modified data to the data store in *Cavanaugh*. Therefore, *Cavanaugh* does not teach or suggest all of the limitations of Claims 1, 15, 26 and 38.

Bannon teaches an object-oriented database (OODB) for providing long-term storage and retrieval of objects created by application programs written at least in part in object-oriented programming languages (see col. 5, lines 39-45). *Bannon* does not teach or suggest "generating automatically, by the processor, an interface within the persistent storage structure, wherein the interface comprises access object classes that are generated automatically by the processor to enable management of object instance data in the persistent storage structure" (emphasis added) as claimed in Claim 1 and essentially as claimed in Claims 15, 26 and 38. As sated in the previous response, consider the following:

Bannon teaches persistent object references (PTR instances) to create, manipulate, store, retrieve, and access persistent objects in the OODB (see col. 9, lines 52-56 and col. 21, lines 55-61). That is, *Bannon* teaches PTR instances to enable management of object instance data in the OODB. As FIG. 2 of *Bannon* illustrates, PTR instances (44) are not located within the OODB (18); rather, they are separate and apart. Thus, *Bannon* does not teach or suggest managing object instance data via an interface located within the persistent storage structure, essentially as claimed in Claims 1, 15, 26 and 38. Compare FIG. 2 of *Bannon* with FIG. 4 of the Application:

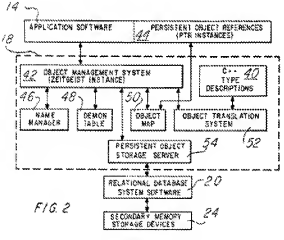

FIG. 2

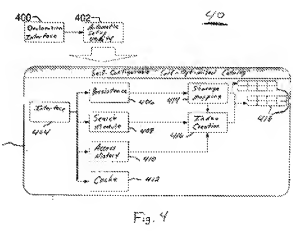

Fig. 4

As shown in FIG. 2, *Bannon* requires an external component (PTR instances (44)) to be interfaced to the OODB (18) in order to manage the object instance data in the OODB. By way of comparison, the interface (404) which manages the object instance data in the persistent storage structure (403) in Claims 1, 15, 26 and 38, is located within the persistent storage structure (403) (see FIG. 4 of the Application). Utilizing a separate, external component located outside of the OODB to manage the object instance data in the OODB, as taught by *Bannon*, is not analogous to an interface within the persistent storage structure enabling management of

object instance data within the persistent storage structure. Thus, *Bannon* fails to cure the deficiencies of *Cavanaugh*.

The combination of *Cavanaugh* and *Bannon* teaches a marshaling process utilizing m-tables to store code which is used for marshaling data and unmarshaling results and persistent object references (PTR instances), located separate and apart from an object-oriented database (OODB), for creating, manipulating, storing, retrieving and accessing persistent objects in the OODB. The combination does not teach or suggest "generating automatically, by the processor, an interface within the persistent storage structure, wherein the interface comprises access object classes that are generated automatically by the processor to enable management of object instance data in the persistent storage structure" as claimed in Claim 1 and essentially as claimed in Claims 15, 26 and 38. Accordingly, the combination does not teach or suggest all of the limitations of Claims 1, 15, 26 and 38.

Therefore, for at least the reasons above, Claims 1, 15, 26 and 38 are believed to be patentable and non-obvious over the combination of *Cavanaugh* and *Bannon*. Applicants respectfully submit that inasmuch as Claims 4-8, 11-14, 18-19, 22-25, 27-28, 30-32, 34-36 and 39 are dependent on Claims 1, 15, 26 and 38, and Claims 1, 15, 26 and 38 are patentable over the cited references, Claims 4-8, 11-14, 18-19, 22-25, 27-28, 30-32, 34-36 and 39 are allowable as being dependent on allowable independent claims. Withdrawal of the instant rejection is respectfully requested.

CONCLUSION

In view of the foregoing, it is believed that all claims now pending patentability define the subject invention over the prior art of record and are in condition for allowance. Early and favorable reconsideration of the case is respectfully requested.

Respectfully submitted,

Date: November 25, 2009

By: /Nathaniel T. Wallace/
Nathaniel T. Wallace
Reg. No. 48,909
Attorney for Applicant(s)

F. Chau & Associates, LLC
130 Woodbury Road
Woodbury, New York 11797
TEL: (516) 692-8888
FAX: (516) 692-8889